8. Appendix

8.1. Defense Mechanism

To briefly reiterate the proposed ODDR mechanism, we make use of a three step process, which helps in outlier detection among the fragments and then we neutralize the same using dimension reduction. This is explained in Figure 1.

Isolation Forest exhibits swift convergence even with a minimal number of trees, and the incorporation of subsampling enhances its ability to yield favorable results while maintaining computational efficiency. The algorithm operates through a recursive process of generating partitions on the dataset, achieved by randomly selecting features and subsequently assigning random split values for these features. An intriguing facet arises from the hypothesis that anomalies necessitate fewer random partitions for isolation compared to their "normal" counterparts in the dataset. Consequently, this characteristic enables Isolation Forest to efficiently and effectively discern anomalies, solidifying its position as a powerful tool for anomaly detection, particularly in datasets with high dimensionality. This is why Isolation Forests were chosen as the base algorithm for the outlier detection task, and then they were modified to suit the specific use-case of finding anomalies in image chunks, as explained in Algorithms 2 and 3.

8.2. Patch-based Attacks

In this evaluation, we employ seven state-of-the-art adversarial patches to rigorously assess model performance. For classification tasks, we utilize the Adversarial Patch (GAP) [4], LAVAN [32], Generative Dynamic Patch Attack (GDPA) [35], and Shape Matters (SM) [13].

Adversarial Patch (GAP) [4] offers a more practical form of attack for real-world scenarios compared to Lp-norm-based adversarial perturbations, which require object capture through a camera. This attack creates universal patches that can be applied anywhere. Additionally, the attack incorporates Expectation over Transformation (EOT) [1] to enhance the strength of the generated adversarial patch.

LAVAN [32] is a technique for generating localized and visible patches that can be applied across various images and locations. This approach involves training the patch iteratively by selecting a random image and placing it at a randomly chosen location. This iterative process makes sure that the model can capture the distinguishing features of the patch across a range of scenarios, thereby enhancing its ability to transfer and its overall effectiveness.

Generative Dynamic Patch Attack (GDPA) [35] is a method that adversarially generates both the patch pattern and its location for each input image. It is presented as a versatile attack framework capable of producing dynamic or static, as well as visible or invisible, patches with minimal configuration adjustments. By utilizing a generator to create both the patch pattern and its location simultaneously for each image, GDPA significantly reduces inference time. Additionally, GDPA can be easily incorporated into adversarial training to enhance a model's robustness against various adversarial attacks.

Shape Matters (SM) [13] proposes a *Deformable Patch Representation (DPR)*, which leverages the geometric structure of triangles to enable a differentiable mapping between contour modeling and masks, allowing the shape to be deformably adjusted during patch generation. Building on DPR, a shape and texture joint optimization algorithm for adversarial patches, termed DAPatch, is introduced. This algorithm effectively optimizes both shape and texture to enhance attack performance. DPR also explicitly examines the importance of shape information on the robustness of deep neural networks (DNNs) from an adversarial perspective, contributing to a deeper understanding of the inherent vulnerabilities of DNNs.

For detection tasks, we use the *AdvYOLO* adversarial patch as introduced in [45]. This method generates a small, compact patch that, when held by an attacker, can effectively deceive the YOLO detector [3]. Typically, deep neural network-based detectors are designed to predict bounding box positions, object probabilities, and class scores for objects within an input image. However, AdvYOLO modifies this process by employing a training approach that minimizes the object probabilities and class scores for the target class—specifically, people. As a result, the detector is tricked into ignoring the presence of individuals, rendering them undetected.

The *Naturalistic Patch* [30] leverages the efficiency of generative adversarial networks (GANs) in crafting physical adversarial patches designed to deceive person detection systems. In their novel approach, Hu et al. [30] focus on manipulating the detection probabilities for the person class. By applying the principles of AdvYOLO, they minimize the objectness and class probabilities specifically targeting the person class, thus creating patches that effectively evade person detectors.

For monocular depth estimation (MDE) models, we employ the *shape-sensitive adversarial patch (SSAP)* as described in [25]. SSAP is specifically designed to disrupt monocular depth estimation systems used in autonomous navigation. By introducing an adaptive adversarial patch, SSAP seeks to mislead depth estimation models, resulting in incorrect depth maps, which can critically affect the performance of autonomous systems. A key characteristic of SSAP is its adaptability to different environments. The patch is optimized through an iterative process that updates the patch using gradients derived from the depth estimation model's backpropagation. This iterative optimization allows SSAP to dynamically adjust and maintain its effectiveness across various scenes, thereby maximizing its ability to deceive the depth estimation model.

8.3. Evaluation Metrics

Here, we briefly mention all the different kinds of evaluation metrics that have been used across this paper to test the performance of ODDR.

8.3.1. Classification Task:

The metric that has been used for the Classification Task is Top-1 percentage accuracy on a test sample of 1000 images. Top-1 percentage accuracy here refers to the proportion of test samples where the model's highest-confidence prediction (i.e., the class with the highest probability score) matches the correct label. In other words, it's the percentage of cases where the model's first prediction is accurate.

8.3.2. Object Detection Task:

To assess the effectiveness of our defense in object detection tasks, we employ the following metrics: *Robust Average Precision* evaluates the model's precision averaged across all recall levels for a given class, particularly under adversarial conditions. This metric quantifies the model's ability to maintain high performance when subjected to attacks, with the defense mechanism in place. *Recovery Rate* quantifies the proportion of correctly restored outputs by the defense mechanism relative to the total number of successful adversarial attacks. This metric captures the inherent positive impact of the defense by measuring its effectiveness in mitigating the adversarial effects.

8.3.3. Monocular Depth Estimation Task:

To evaluate the effectiveness of our proposed defense, we utilize the same metrics as those used in [25]: the *mean depth estimation error* (E_d) associated with the target object, and the *ratio of the affected region* (R_a). For the computation of these metrics, the depth prediction of the clean target object is considered the ground truth.

The mean depth estimation error quantifies the extent to which our proposed adversarial patch impacts the accuracy of depth estimation. A higher value in this metric suggests a more successful attack. Similarly, the ratio of the affected region indicates the extent of the attack's influence, with higher values signifying a more impactful attack.

The *mean depth estimation error* is calculated as follows:

$$E_d = \frac{\sum_{i,j}(|d - d_{adv}| \odot M_f)}{\sum_{i,j} M_f} \tag{3}$$

where d represents the clean predicted depth, and d_{adv} is the depth prediction after the adversarial attack.

The *ratio of the affected region* measures the percentage of pixels whose depth values are altered beyond a specific threshold, in relation to the total number of pixels within the focus mask (M_f). Pixels with depth changes greater than 0.1 are considered affected. This metric is calculated as follows:

$$R_a = \frac{\sum_{i,j} I((|d - d_{adv}| \odot M_f) > 0.1)}{\sum_{i,j} M_f} \tag{4}$$

Additionally, we use the Mean Square Error (MSE) to evaluate the model's performance concerning the predicted depth map from an unperturbed input. The MSE is computed as:

$$MSE = \frac{1}{N}\sum_{i,j}(d_{adv_{i,j}} - d_{i,j})^2 \tag{5}$$

where N represents the total number of pixels.

8.4. Additional Results for Classification Task

Here, we present findings demonstrating the efficacy of our defense mechanism against two newly introduced attack methodologies, across an expanded array of model architectures, including other CNNs and vision transformer models. Additionally, we explore the impact of different hyper-parameters on defense performance. Table 2 presents ODDR's performance against two attacks: Generative Dynamic Patch Attack (GDPA) [35] & Shape Matters (SM) [13]. For Detection tasks, apart from YOLO, we further tested the Faster-RCNN model on a subset of INRIA dataset, which generated a clean accuracy of 100%, adversarial accuracy of 33.33% and post ODDR robust accuracy of 86.66%.

As mentioned in the Results section, our evaluation was focused on measuring the model's robust accuracy. To illustrate the impact of our defense strategy, we initially generated adversarial patches using two distinct attack strategies, namely LAVAN, GoogleAp and GDPA. Subsequently, we reported the model's robust accuracy across different patch sizes and various models, for LAVAN and GoogleAP here, due to restrictions of content volume, the complete set of results including all the patch sizes. It may also be noted here that the performances of ODDR on clean samples without the patches are 79.8% (ResNet-152), 77.3% (ResNet-50) and 73.2% (VGG-19) on ImageNet and 92.8% (ResNet-152), 88.9% (ResNet-50) and 87.1% (VGG-19) on CalTech-101.

Tables 8 and 9 highlight the fact that our proposed ODDR is able to maintain impressive performance across different degrees of potency of attacks, showcasing a remarkable level of robust accuracy when countering GoogleAp and LAVAN attacks on the ImageNet dataset. For instance, our defense achieves robust accuracy rates of 79.1% and 74.1% when employed against GoogleAp and LAVAN attacks, respectively for a smaller patch size of 38x38 pixels and 80.3% and 75.4% on the largest of patch sizes of 50x50 pixels.

Patch Size	Model / Neural Network	Baseline Accuracy	Adversarial Accuracy	Robustness (w/ patch)
38	ResNet 152	81.2%	39.9%	79.1%
x	ResNet 50	78.4%	38.8%	75.6%
38	VGG 19	74.2%	39.1%	72.8%
41	ResNet 152	81.2%	21.4%	79.6%
x	ResNet 50	78.4%	21.1%	77.1%
41	VGG 19	74.2%	22.8%	71.9%
44	ResNet 152	81.2%	14.6%	80.1%
x	ResNet 50	78.4%	14.2%	76.4%
44	VGG 19	74.2%	15.8%	72.1%
47	ResNet 152	81.2%	9.3%	78.9%
x	ResNet 50	78.4%	9.0%	77.2%
47	VGG 19	74.2%	10.6%	72.3%
50	ResNet 152	81.2%	4.9%	80.3%
x	ResNet 50	78.4%	4.5%	77.8%
50	VGG 19	74.2%	3.8%	72.5%

Table 8. ODDR robustness on GoogleAp attack (ImageNet dataset)

Patch Size	Model / Neural Network	Baseline Accuracy	Adversarial Accuracy	Robustness (w/ patch)
38	ResNet 152	81.2%	10.1%	74.1%
x	ResNet 50	78.4%	10.2%	70.2%
38	VGG 19	74.2%	11.1%	71.1%
41	ResNet 152	81.2%	7.9%	76.9%
x	ResNet 50	78.4%	8.3%	74.7%
41	VGG 19	74.2%	8.1%	70.1%
44	ResNet 152	81.2%	4.9%	76.3%
x	ResNet 50	78.4%	4.8%	72.8%
44	VGG 19	74.2%	4.8%	73.1%
47	ResNet 152	81.2%	1.2%	78.4%
x	ResNet 50	78.4%	1.0%	77%
47	VGG 19	74.2%	1.7%	72.1%
50	ResNet 152	81.2%	1.9%	75.4%
x	ResNet 50	78.4%	2.0%	74.1%
50	VGG 19	74.2%	2.1%	71.8%

Table 9. ODDR robustness on LAVAN attack (ImageNet dataset)

Exactly as in Section Experimental Results, the extended versions are presented in Tables 10 and 11 for the Caltech-101 dataset. As evident, our proposed ODDR defense technique achieves outstanding performance with robust accuracy rates of 90.8% and 91.1% when defending against GoogleAp and LAVAN attacks, respectively for a smaller patch size of 38x38 pixels and 91.3% and 91.6% on the largest of patch sizes of 50x50.

The overall takeaway from the extensive experimentation is that unlike many other defense techniques available in the literature, the proposed ODDR mechanism is able to successfully thwart the patch based adversarial attacks irrespective of the patch sizes. Table 12 presents a comparative study of ODDR's robustness compared to many state-of-the-art defenses against GAP [4] attack with the ResNet-50 model on the ImageNet dataset.

Patch Size	Model / Neural Network	Baseline Accuracy	Adversarial Accuracy	Robustness (w/ patch)
38	ResNet 152	94.1%	48.6%	90.8%
x	ResNet 50	90.9%	49.2%	86.4%
38	VGG 19	88.6%	47.1%	85.6%
41	ResNet 152	94.1%	30.1%	91.2%
x	ResNet 50	90.9%	29.3%	87.1%
41	VGG 19	88.6%	28.2%	87.3%
44	ResNet 152	94.1%	10.8%	90.2%
x	ResNet 50	90.9%	11.2%	86.9%
44	VGG 19	88.6%	12.6%	85.4%
47	ResNet 152	94.1%	9.1%	91.2%
x	ResNet 50	90.9%	9.6%	86.5%
47	VGG 19	88.6%	10.4%	86.2%
50	ResNet 152	94.1%	6.8%	91.3%
x	ResNet 50	90.9%	5.9%	87.2%
50	VGG 19	88.6%	6.2%	85.4%

Table 10. ODDR robustness on GoogleAp attack (CalTech-101 dataset)

Patch Size	Model / Neural Network	Baseline Accuracy	Adversarial Accuracy	Robustness (w/ patch)
38	ResNet 152	94.1%	15.6%	91.1%
x	ResNet 50	90.9%	17.1%	87.3%
38	VGG 19	88.6%	15.3%	84.9%
41	ResNet 152	94.1%	14.2%	90.9%
x	ResNet 50	90.9%	13.9%	86.8%
41	VGG 19	88.6%	13.8%	85.8%
44	ResNet 152	94.1%	8.4%	91.3%
x	ResNet 50	90.9%	8.9%	87.8%
44	VGG 19	88.6%	9.1%	84.8%
47	ResNet 152	94.1%	5.1%	90.8%
x	ResNet 50	90.9%	4.9%	88.1%
47	VGG 19	88.6%	6.1%	86.4%
50	ResNet 152	94.1%	1.2%	91.6%
x	ResNet 50	90.9%	1.0%	86.7%
50	VGG 19	88.6%	1.8%	85.6%

Table 11. ODDR robustness on LAVAN attack (CalTech-101 dataset)

Defense Methods	Robust Accuracy
Localised Gradient Smoothing [40]	53.86%
Jujutsu [14]	60%
De-Randomised Smoothing [33]	35.02%
FNC [48]	59.6%
PatchGuard [47]	30.96%
Anomaly Unveiled [9]	67.10%
Dimensionality Reduction [8]	66.2%
ODDR (Ours)	**75.65%**

Table 12. Performance of our proposed defense compared to many state-of-the-art defenses against GAP [4] attack with the ResNet-50 model on the ImageNet dataset.

8.5. Hyper-parameter Tuning

As mentioned earlier, ODDR uses two categories of hyper-parameters: *Active hyper-parameters* and *Passive hyper-parameters*. The active hyper-parameters significantly influence the performance of ODDR, and the results reported in the tables correspond to their optimal tuned values. These include c (the confidence level for identifying anomalies, ranging from 0.8 to 0.95) and inf (the information preserved after SVD, ranging from 0.7 to 0.9). Figure 7 shows the change in the robust accuracy of ODDR with different hyper-parameters for the classification task, on three different neural network models. It is note-worthy that there is a knee-bend in the plots, and selecting any value before that point of bending, within a range (denoted within red lines), works equally well.

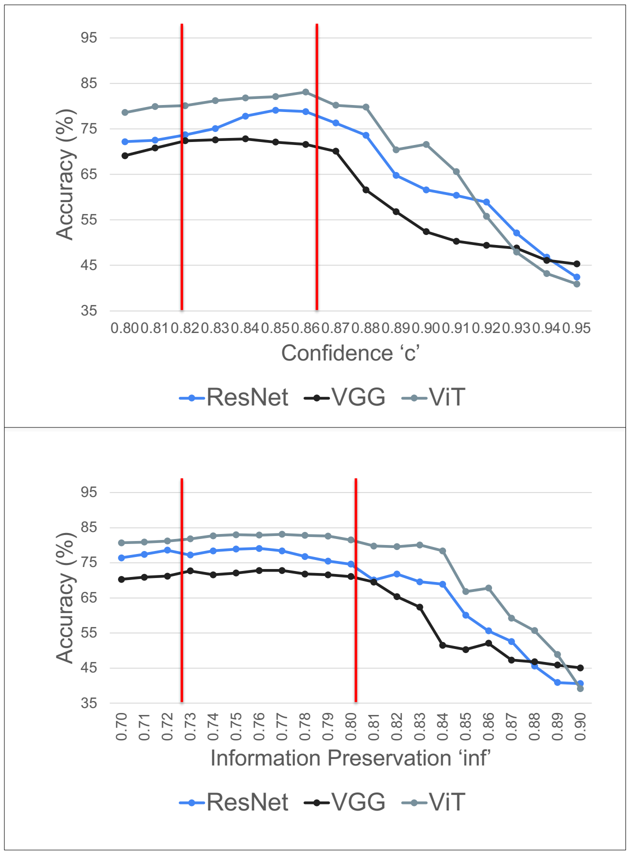

Figure 7. Plots of hyper-parameter tuning for ODDR. The two *Active Hyper-parameters* are c (the confidence level for identifying anomalies, ranging from 0.8 to 0.95) and inf (the information preserved after SVD, ranging from 0.7 to 0.9).

8.6. Adaptive Attack details

Here we elaborate on the detailed mechanism of the implementation of the adaptive attacks to test the robustness of ODDR.

8.6.1. Distribution Mapping

In order to generate a patch capable of circumventing our defense, the adversarial noise must adhere to a distribution similar/close to that of clean images. We implement the adaptive attack by constraining the distribution of the adversarial patch to closely match the average distribution of n randomly selected fragments from the clean image. Specifically, we calculate the average distribution in terms of both mean and standard deviation. We then compute the mean difference and standard deviation ratio between the average distribution of the fragments and the adversarial patch. This helps us in generating an attack which is able to counter the premise of the ODDR defense mechanism.

Taking a step back to study whether the adversarial noise contained in the adversarial patches belong to a different distribution or not, as compared to the clean images, we make use of a distance metric to measure the distance between the overall distribution of the image and the adversarial patch. Specifically, we split the adversarial image along with the patch into fragments (as described in Section ODDR Defense Mechanism and fit a distribution to it, and calculate the Mahalanobis distance of every fragment from that distribution [41]. The Mahalanobis distance [16] is a distance metric which is designed to measure the distances of data points with respect to a distribution. Formally, for a probability distribution Q on $\mathbb{R}$, which has the mean $\mu = (\mu_1, \ldots, \mu_N)^T$, a positive definite covariance matrix S, then for any point $x = (x_1, \ldots, x_N)^T$ for the said distribution Q, the Mahalanobis distance is [39]:

$$d_M(x, Q) = \sqrt{(x-\mu)^T S^{-1} (x-\mu)}$$

For our case, the x_is are the fragments created as part of the Fragmentation phase. Once we calculate the Mahalanobis distance for every such fragment, and plot them, we observe a bi-modal distribution in Figure 8, with the distance measured for the fragments lying on the adversarial patch having a significantly higher value of Mahalanobis distance than the rest, belonging to a different distribution. This is a clear indication that the patch contains informational variability that is different from the rest of the image and can be isolated as anomalies.

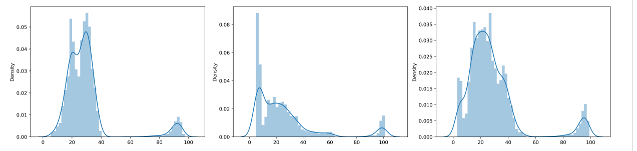

Figure 8. Plot of Mahalanobis distances of fragments to represent anomalous behaviour of adversarial patches as seen in the bi-modal distribution.

Once we generated the adversarial patches using the adaptive attack proposed in Section Adaptive Attack in the main paper, we recalculated the Mahalanobis distance for each fragment and plotted them. Upon observation, we noticed a one single heavy tailed distribution indicating that the fragments located on the adversarial patch exhibited a Mahalanobis distance closer to that of the rest of the image fragments (See Figure 9) which makes it challenging to isolate the patch as anomalies.

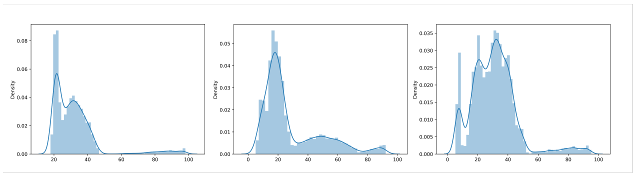

Figure 9. Plot of Mahalanobis distances of fragments upon introducing the adaptive attack, showing one single heavy tailed distribution.

The key takeaway from this exercise is to showcase the fact that adversarial patches contain information that do not belong to the distribution of the information contained in the rest of the images or video frames. This is precisely why the outlier detection mechanism is successful in isolating the fragments which contain the adversarial patch. If however, we do force the adversarial patch to contain information belonging to the distribution of the rest of the image, by setting constraints while training the patch, as in our proposed adversarial attack in the Adaptive Attack Section , then the patch itself becomes less effective and is unable to bring down the accuracy of the classifier or object detector.

8.6.2. Zeroth Order Gradient based Attack

These methods estimate gradients using black-box optimization techniques instead of relying on explicit backpropagation. Given a classifier $f : \mathbb{R}^d \rightarrow \mathbb{R}^K$ mapping an input $x \in \mathbb{R}^d$ to a probability distribution over K classes, we aim to find an adversarial patch P such that:

$$\arg\max_i f(x + M \odot P) \neq y, \tag{6}$$

where:

- y is the true label of x.
- $P \in \mathbb{R}^{m\times n\times c}$ is the adversarial patch of size $m \times n$ with c color channels.
- $M \in \{0, 1\}^d$ is a binary mask indicating the patch location.
- $\odot$ represents element-wise multiplication.

Since direct gradient access is unavailable, we approximate gradients using finite differences:

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x + \delta e_i) - f(x - \delta e_i)}{2\delta},$$

where:

- e_i is a standard basis vector perturbing the i-th pixel in the patch.
- δ is a small finite difference step.

The adversarial objective is to maximize the classifier's loss:

$$\max_P \mathcal{L}(f(x + M \odot P), y),$$

where $\mathcal{L}$ is the adversarial loss function (e.g., cross-entropy loss).

The patch update follows a gradient ascent step:

$$P^{(t+1)} = P^{(t)} + \alpha\nabla_P\mathcal{L},$$

where α is the step size.

Using the zeroth-order gradient approximation, the gradient estimate is ($\nabla_P\mathcal{L} \approx$):

$$\frac{\mathcal{L}(f(x + M \odot (P + \delta u)), y) - \mathcal{L}(f(x + M \odot (P - \delta u)), y)}{2\delta} u,$$

where u is a random perturbation vector. The algorithm for generating the adversarial patch using the Zeroth Order Gradient based scheme is mentioned here.

Algorithm 4 Zeroth-Order Adversarial Patch Attack

```
IN Classifier $f$, Input Image $x$, True Label $y$, Patch Mask $M$, Learning Rate $\alpha$, Finite Difference Step $\delta$, Maximum Iterations $T$
Initialize adversarial patch $P \sim U(0, 1)$
For $t = 1$ to $T$
Sample random perturbation vector $u \sim N(0, I)$
Compute zeroth-order gradient estimate:
$\nabla_P\mathcal{L} \approx \frac{\mathcal{L}(f(x+M\odot(P+\delta u)),y)-\mathcal{L}(f(x+M\odot(P-\delta u)),y)}{2\delta} u$
Update patch using gradient ascent:
$P \leftarrow P + \alpha\nabla_P\mathcal{L}$
Project $P$ onto valid pixel space $[0, 1]$
Return Adversarially optimized patch $P$
```

8.6.3. Lower Dimensional Patch

The objective here is to test if it is possible to generate a patch that is immune to the process of dimension reduction, which is the technique used by ODDR for neutralizing the patch. To accomplish this, during the training of the patch, we introduced an additional step of performing dimension reduction using SVD (with a minimal 1% drop in informational content) on the adversarial patch in every iteration. We have noted that upon subjecting the patch to dimension reduction, the patch loses its adversarial property. This is because every time the adversarial patch is processed through dimension reduction, the SVD algorithm projects the sample on a different orthonormal basis and expresses its variability within it as a linear combination of component vectors. Since this underlying basis is not identical, this transformation can not be used in an adaptive manner to generate an immune adversarial patch. This makes dimension reduction using Singular Value Decomposition a very powerful method of neutralizing the detected adversarial patch.

8.7. Sample Images

We have tested ODDR for its functionality on various machine learning tasks, datasets and adversarial patches. Some of the samples are presented here. Figure 10 shows how ODDR is able to identify and neutralize the adversarial patch for Image Classification Task, on the ImageNet Dataset [17], using the ResNet-50 Model [28] and Google Adversarial Patch [4].

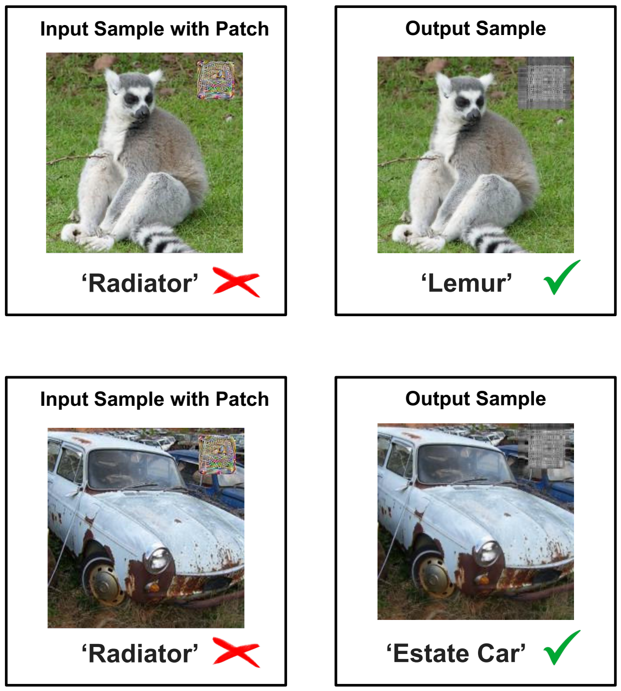

Figure 10. Samples for testing ODDR in action: Task: Image Classification, Dataset: ImageNet [17], Model: ResNet-50 [28], Adversarial Patch: Google Adversarial Patch [4]

Figure 11 shows how ODDR is able to identify and neutralize the adversarial patch for Person Detection Task (which is a sub-task of the generic Object Detection Task), on the INRIA [15] Dataset , using the YOLO [3] Model and AdvYOLO [45] patch. It is to be noted here that upon the introduction of the patch on the initially detected 'person' in the clean sample, the task fails subsequently. However, upon subjecting the sample with the patch to the ODDR framework, the person detection task again is successful.

Figure 12 shows how ODDR is able to identify and neutralize the adversarial patch for Object Detection Task, on the INRIA [15] Dataset , using the YOLO [3] Model and AdvYOLO [45] patch. We would like to highlight here that when the patch is applied to the detected 'person' in the clean image, all other objects except the person is detected by the model. The detection of the person and the other objects is successful again upon processing the sample with ODDR.

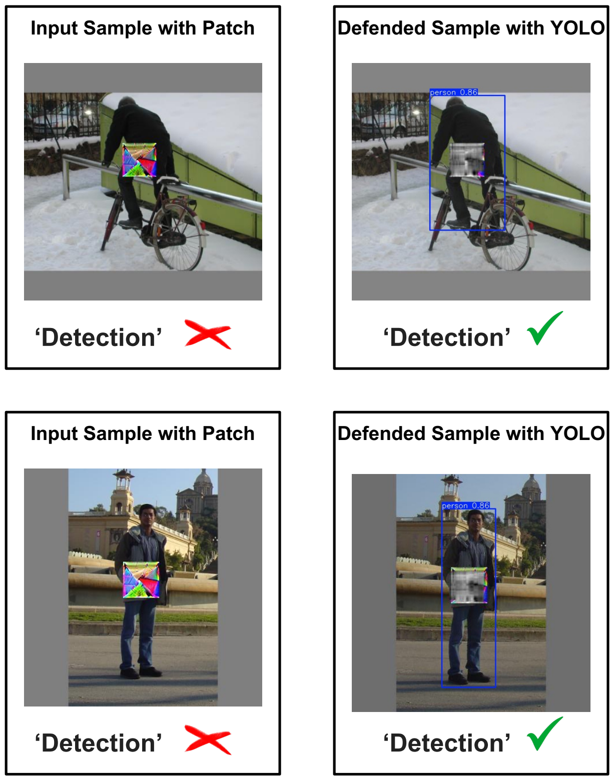

Figure 11. Samples for testing ODDR in action: Task: Person Detection (Object Detection sub-task), Dataset: INRIA [15], Model: YOLO [3], Adversarial Patch: AdvYOLO [45].

8.7.1. Samples for Inspection

It is not possible to include all samples as part of the supplementary materials, so a randomly chosen few are added to demonstrate the technique and substantiate the findings. The sample images have been attached in the folder contained within the supplementary materials. The submitted folder should be navigated as mentioned hereafter.

The root directory is called *SAMPLES*, which contains three folders, for the three types of machine learning applications that we have considered, namely *IMAGE CLASSIFICATION*, *OBJECT DETECTION* and *MONOCULAR DEPTH ESTIMATION*. Within the *CLASSIFICATION* folder, there are two sub-folders, one containing the *ImageNet_samples* (the naming of the individual files are self-explanatory) and the other containing the samples from *GradCam*, which has been used to understand model inter-

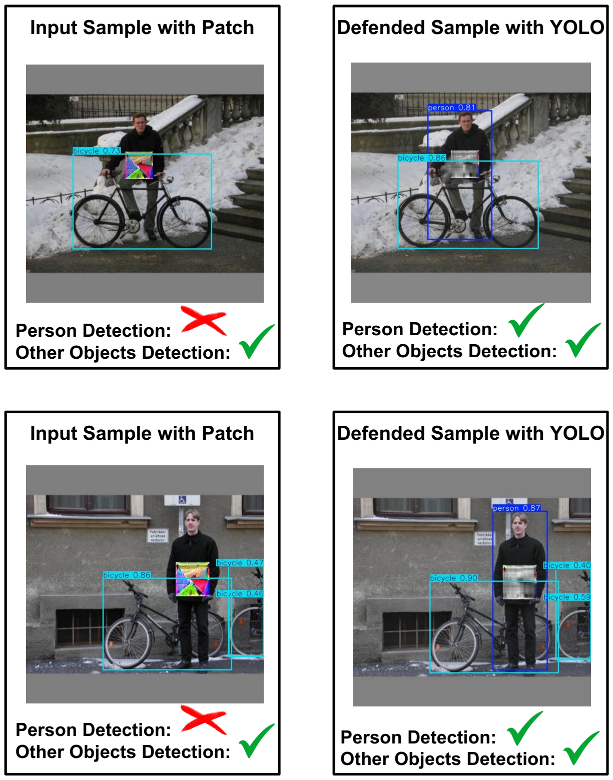

Figure 12. Samples for testing ODDR in action: Task: Object Detection, Dataset: INRIA [15], Model: YOLO [3], Adversarial Patch: AdvYOLO [45]

pretability in Section Discussion. Within the *DETECTION* folder, we have two sub-folders corresponding to the two datasets that we have used in the detection task. Each of them, *CASIA* and *INRIA* contain two sub-folders for two different types of adversarial patches that have been used in the experiments in the earlier sections. For each combination of the dataset and adversarial patch, we have the folders containing *samples_with_ODDR* (that have the adversarial samples after having undergone the ODDR defence pipeline) and the *inference* which show the efficacy of the ODDR scheme by running the inference model on the ODDR-ed samples. The samples which have undergone the ODDR based defense technique could be tested with the correspondingly appropriate inference model as mentioned in the experiments section for verification. Within the *DEPTH ESTIMATION* folder, there are three sub-folders corresponding to the *Clean Samples*, *Adversarial Samples* where the patch is inserted and the *Defended Samples* which have ODDR implemented on them. The corresponding depth map is also provided for each sample in the respective folders.

8.8. Code

This work is supported by a project grant which marks the code as closed IP and proprietary and therefore it is something that cannot be shared in the public domain unfortunately.

8.9. Working Demo

As part of the supplementary materials, we have included a video demonstration of ODDR in action. To maintain anonymity we have used an open source dataset CASIA [49] for the same. The two videos, for the adversarial patch attack and the defended version using ODDR are included in the DEMO folder. Specifically, the video titled *AdversarialPatchVideo.mp4* is the one where the AdvYOLO patch is applied on the samples with YOLO being run on it, and the video titled *DefendedVideo.mp4* is the one where the samples have been introduced to ODDR and the YOLO run on it. Apart from the videos, the folder also contains the individual samples/frames in a separate sub-folder for reference.